Reinforcement Learning: Final Exam May 15, 2020

Name: _____

Start time: _____ Finish time: _____ Total minutes_____

Answer the questions in the answer spaces provided on the question sheets. If you run out of room for an answer, note in the answer space that it is continued on another page, and continue on a blank sheet.

No use of a computing device is allowed other than as a timer, clock, music player, simple (non-programmable) scientific calculator (for example, https://www.calculator.net/scientific-calculator.html), or for word processing (editing this PDF file or writing your answers in some word processor). This is a closed-book exam. You may have one 8.5x11 double-sided handwritten page of notes.

If you think something about a question is open to interpretation, write any assumptions you've made as part of answering the question.

Be concise in your answers; you need not try to fill in all or even most of the space provided for an answer. Show your work, though, since partial credit may be awarded.

You have four contiguous hours to complete this exam starting from when you look at any page other than the first. The four hours is more than I think you need, but should allow for any needed time for dealing with computer issues in creating your final PDF.

Submit the completed exam to Gradescope no later than 5 PM, May 15, 2020. You may submit:

- An edited version of the PDF with answers added (PDF editor on iPad for example, with handwritten answers, or Preview on Mac with text annotation).
- A scanned paper version of the PDF (that you've handwritten answers on). Make sure the scanned version is legible before you submit it.
- A PDF output of some word processor. For example, you can write your answers in LaTeX, or Microsoft Word (with Equations), or Google Docs (with Auto-Latex addon). You need not repeat the question: just provide your answer.

In Gradescope, please make sure to assign each question to the page(s) that contain your answer. It's much easier to grade that way.

There are 100 points on the exam.

Good luck!

8 points 1. What is a multi-armed bandit problem?

8 points 2. Give two methods for dealing with a non-stationary (changes over time) multi-armed bandit environment.

3. Your Monte-Carlo algorithm generates the following episode using policy $\pi$ when interacting with its environment. This is the first episode that has been generated.

| Timestep | Reward | State | Action |
|---|---|---|---|
| 0 | | S1 | A1 |
| 1 | 13 | S1 | A2 |
| 2 | 7 | S1 | A1 |
| 3 | 13 | S1 | A2 |
| 4 | 14 | T | |

Assume the discount factor, $\gamma$, is $\frac{1}{2}$.

9 points (a) What are the estimates of: $q_\pi(S1, A1)$ and $q_\pi(S1, A2)$ if using first-visit?

9 points (b) What are the estimates of: $q_\pi(S1, A1)$ and $q_\pi(S1, A2)$ if using every-visit?

4. Your off-policy Monte-Carlo algorithm generates the following episode using policy $b$ when interacting with its environment.

| Timestep | Reward | State | Action |
|---|---|---|---|
| 0 | | S1 | A1 |
| 1 | 5 | S2 | A2 |
| 2 | 12 | S3 | A1 |
| 3 | 6 | T | |

Use this single episode to estimate the value of state $S1$ following policy $\pi$. Assume the discount factor, $\gamma$, is 1.

Here are the two policies:

| | A1 \| S1 | A2 \| S1 | A1 \| S2 | A2 \| S2 | A1 \| S3 | A2 \| S3 |
|---|---|---|---|---|---|---|
| $\pi(A\|S)$ | $\frac{1}{2}$ | $\frac{1}{2}$ | 0 | 1 | $\frac{1}{2}$ | $\frac{1}{2}$ |
| $b(A\|S)$ | $\frac{1}{4}$ | $\frac{3}{4}$ | $\frac{1}{2}$ | $\frac{1}{2}$ | $\frac{1}{4}$ | $\frac{3}{4}$ |

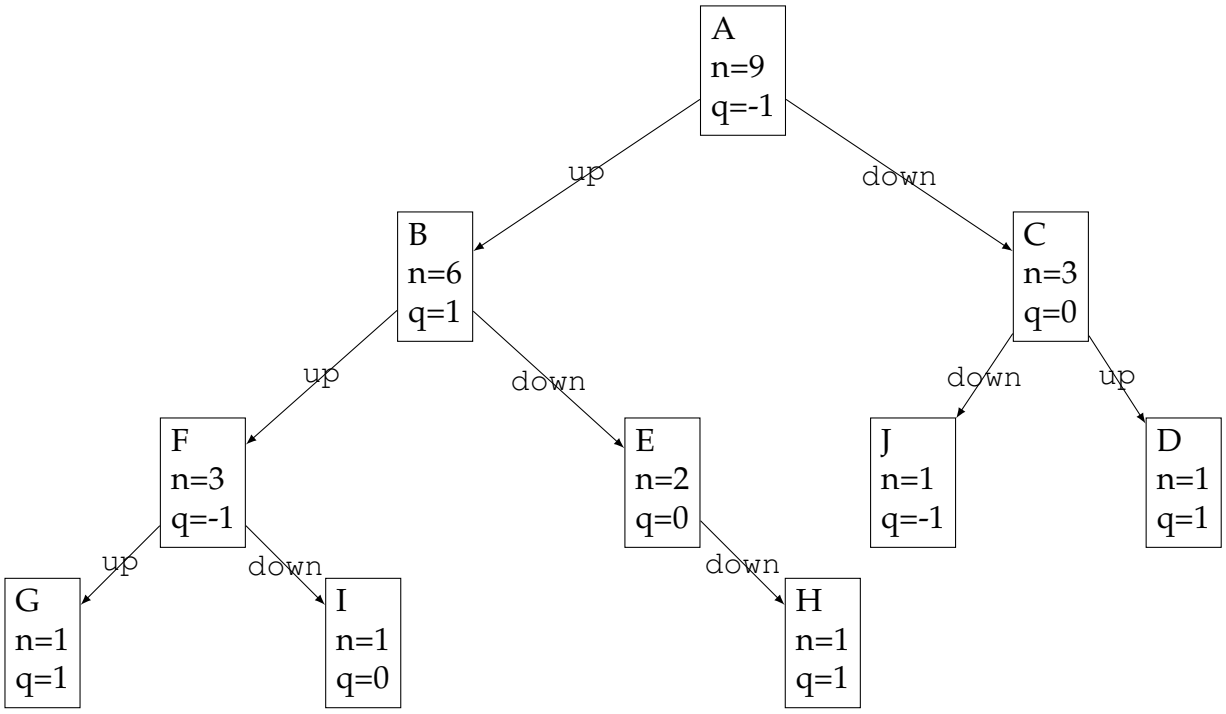
9 points

(a) What are the estimates of: $v_\pi(S1)$ and $q_\pi(S1, A1)$ if using ordinary importance sampling (and first-visit)?

9 points

(b) What is the estimate of: $v_\pi(S1)$ if using per-decision importance sampling (and first-visit)?

5. You are using Monte Carlo Tree Search to decide on the next action for a two-person competitive game with 2 actions at each state (`up` and `down`). It is player 1's turn to play in state A. The state of the tree so far is as follows (each node consists of state identifier, n value, and q value):




Recall that the formula for the UCT value for a node, $v$, is:

$$UCT(v) = \frac{q(v)}{n(v)} + c\sqrt{\frac{\ln n(v.parent)}{n(v)}}$$

Assume the constant $c$ in the UCT formula is 0.5.

7 points (a) What is the node that is next selected (show your work)?

3 points (b) What are the complete contents of the node that is expanded from the selected node?

3 points (c) Assuming that the simulation (rollout) from the expanded node gives a value of 1 (that is, player 1 wins), backup that value to all of the affected nodes and provide the new values for each such node.

3 points (d) Assume that after this final rollout, we've run out of time to run the MCTS simulation and must now choose an action for player 1 from state A. What action will be chosen and why?

8 points 6. Direct reinforcement learning updates a *policy* based on *interactions with the environment*. Planning (indirect reinforcement learning), however, updates a _____ based on _____.

8 points 7. Why should the action chosen by Monte Carlo Tree Search tend to be better than the action the underlying rollout policy would choose?

8 points 8. Monte Carlo methods for learning value functions require episodic tasks. Why, specifically? How is it that n-step TD methods avoid this limitation and can work with continuing tasks?

8 points 9. MuZero learns to play chess, Go, and Atari games without explicitly knowing the rules/dynamics. One of the difficulties with model-based RL is that the model may be close-to-correct for a single time period but diverges over time. How, specifically, does Mu-zero reduce this divergence?